Universal Location Referencing: A New Approach for Dynamic Location Referencing in TPEG

Progress Report, TPEG-LOC2 Study

Fraunhofer FIRST
January 13, 2012

Authors:
Andreas Schramm
Birgit Kwella
Matthias Schmidt
Norbert Pieth
Thilo Ernst

Fraunhofer FIRST, Kekuléstraße. 7, 12489 Berlin, Germany

Table of Content

Table of Figures

About TPEG

TPEG (Transport Protocol Experts Group) [1] is a protocol specification for the exchange of RTTI (Real Time Traffic Information) to provide road users with comprehensive, up-to-date traffic and traveller information across multiple transport modes, thus allowing door-to-door support. TPEG consists of a collection of ISO/TS standards that already cover a number of essential applications, and is extensible through new applications. It is being constantly developed further by the TISA (Traveller Information Services Association) [2]. TISA is a non-profit organization with more than 100 members from industry and public institutions.

TPEG has the aim to describe real-world events and situational data with their spatio-temporal scope, i.e. the geographical location and time or time period they are associated with. Those (event and situational data) originate in established sensor, control and data management infrastructures e.g. for municipal traffic management, public transport operation and weather observation. Some of these data is currently distributed to vehicle-based and other mobile client devices through the widely-deployed RDS-TMC system [4]. However, this is only a small subset of the RTTI information available today because of severe limitations of TMC [3], which in turn resulted from the very low transport capacity of its original carrier RDS. To overcome these limitations TPEG was created. Beside the traditional but limited scope of TMC TPEG addresses new application areas, e.g. public transport, weather, current traffic flow and prediction, parking information (and more) in the form of TPEG application specifications. Furthermore it remains extensible through additional future applications and a process for their specification within the TISA and their representation in the form of standards or specifications [6][7]. All TPEG applications need provisions to locate the event or situational data they express precisely in their corresponding geographical region.

TPEG is very flexible, so a wide range of client devices is possible and these devices may be designed for very different scenarios. A pure text-based client system may support an application within the public transport sector; a text-to-speech system may concentrate on assistance for handicapped users and a powerful navigation system may cover nearly all defined applications. Different devices and scenarios will impose different requirements w.r.t. to the form and accuracy of associated location information: For a radio with basic TPEG support and a simple, textual display, support for a short natural-language description of a location will be sufficient; in a navigation system with comprehensive on-board maps it is essential to associate each piece of TPEG information with exactly the right road segment or other elements from the locally installed map in order to allow the system to take optimal decisions in route planning and guidance.

TPEG Structure and ULR Motivation

In the TPEG specification three major containers hold all necessary information to manage the information flow of RTTI messages (that express events and situational data) from producer side to consumer[1] side. The management container holds all essential temporal and administrative information to manage each message during its lifetime. The event container holds all information to describe the event itself and further details of it. The location referencing

[1] in TPEG *producer* is used synonymously to *server* and *consumer* synonymously to *client* as the standard usage scenario is infrastructure-to-vehicle

container holds all spatial information to describe the geographical location and scope of this event; more concretely, it may hold one or several location references (all of which describe the same conceptual location, only using different location referencing methods). The TPEG protocol structure is defined in a way that a client may concentrate on certain applications and containers and can easily skip messages which are of no interest to the client. Unknown elements in the input stream (e.g. messages belonging to TPEG applications that were not yet defined or were disregarded at the time the client software was implemented) will be skipped as well; the same holds true for the different types of location references.

Several types of location references (and the location referencing methods used to encode and decode them) already are specified as standards or as drafts in TPEG[2].

The remainder of this document concentrates on one (new) location referencing method and type named TPEG-ULR (Universal Location Referencing), which aims to overcome the limits of TPEG-LOC [5] (in terms of efficiency and accuracy) based on an open, royalty-free method. Another essential goal of ULR is to provide a location referencing scheme that is both human-readable and machine-processable, and supports basic functionality even if the client system does not have an on-board digital map. An ULR location reference contains geo-graphical co-ordinates according to the WGS84 standard and additional information such as direction and height or level if applicable. The goal is to always provide enough information to have realistic chances to reconstruct the location with sufficient accuracy on the consumer side so the user or system on the client-side can correctly react on the message. Some parameters are mandatory (i.e. will always be included by the producer) and some others are optional. The consumer side can always rely on the mandatory part and can profit from the optional parameters if they are present and the client system is capable of processing them.

ULR Generation Algorithm

The further descriptions will handle the producer side and actions to build an ULR reference from original location information. The original location information could e.g. be:

- A database entry contain geographical co-ordinates, e.g. from a database of parking areas
- Approximate geographical co-ordinates from an interactive system in which an operator uses a clickable map to select a location
- A pre-generated TPEG location reference which is retrieved from a database or another means of storage, or received from another system through a communication channel
- A known road segment or other map feature, or set thereof, in the source map[3]

In case some of these original location data might contain co-ordinates which cannot be guaranteed to have direct counterparts in the source map, a preparation step can be executed which determines the most appropriate source map features corresponding to these co-

[2] Apart from the TPEG-LOC format to be superseded by ULR, the most important types are TMC/ALERT-C location references and DLR1 (AGORA-C) location references. Another open alternative called OpenLR has not been introduced to the TPEG standardization up to now.

[3] source map: digital map used on the producer (server) side

ordinates. Also co-ordinates defined in non-WGS84 systems will be converted to WGS84. Details of this process are outside the scope of this document.

At this stage WGS84 co-ordinates and corresponding source map features for the location are known, which also makes some additional attributes such as a height value available. All further parameters that ULR uses to characterize the location will be determined in the process described below, e.g. direction, distances, descriptive values, textual values for the human readable names etc.

It is possible and indeed the rule rather than the exception that the destination map[4] is different from the source map, e.g. because of map producer or map version. Furthermore resource-constrained clients may use lossy compression techniques to reduce the map's storage requirements, introducing additional inaccuracies. Due to all these factors, identifying the actual location within the consumer-side on-board map is a nontrivial task. As an example a particular road segment not only might have other end-point co-ordinates than in the source map, but the system must also be able to cope with the case that due to topological differences, the "same" segment does not exist in the target map at all, so another "close" segment must be chosen in a reasonable way. To achieve this, additional parameters are introduced for a more detailed description of the location. Some are distance-based and for more complicated map deviations a special algorithm based on Markov chains[5] is employed.

Element Overview

ULR supports three types of locations:

1. Point location for single point events
2. Linear location for events placed on single segments or polylines
3. Area location for affected areas of defined coverage

ULR supports the following parameters (mandatory and optional ones, single values or data sets):

1. WGS84 co-ordinate
2. Direction
3. Category (e.g. street category)
4. Height/Level
5. Distance
6. Relations between segments
7. Accuracy
8. Language
9. Text
10. ... further descriptive parameters, e.g. out of a set of table entries

Strategy

The following section will sketch an algorithm of mandatory and optional actions to fill the ULR location container. It is reduced to the actions necessary to support the machine processable

[4] destination map: digital map used on the consumer (client) side
[5] ULR Markov algorithm, described later on

part, i.e. the filling of text-valued parameters with natural language text is not further discussed here. Generally it can be stated that these human interface focussed decisions and actions will follow after the described actions for the machine handling herein, e.g. inclusion of the street name of the identified segment. Details are subject to further modifications and refinement.

Determining Mandatory Parameters

- Selection of the location type in accordance to the event
  - Use exactly one of the location types:
    - Point
    - Linear
    - Area
- For Point:
  - Place the point's WGS84 co-ordinates as found on the source map into the ULR data set
- For Linear:
  - Definition with a logical analysis of the descriptive intent *from, to, via*
    - Determine the start point on the source map
    - Determine the end point on the source map
    - If applicable: Determine an ordered sequence of intermediate (via) points between start and end on the source map, beginning from start
  - For each start, via, and end point place its co-ordinates as found in the source map in the ULR data set in the ordered way mentioned above
  - For each segment start point, as far as applicable, identify the direction of the segment and place the direction value in the ULR data set
  - For each segment start point, as far as applicable, identify if a functional relation[6] to the preceding segment exist and mark this in the ULR data set
- For Area:
  - Definition to distinguish between simple areas and complex areas
    - For a simple area location: Map of the centre point to WGS84 co-ordinates on the source map and place its co-ordinates in the ULR data set
    - For a complex area location: Build a surrounding polygon with start point and at least two ordered intermediate points and place these co-ordinates in the ULR data set

Determining Optional Parameters

The optional parameters will be used if applicable. They should be used if it can be assumed that they support a significantly better interpretation and matching on the consumer side. So some decisions must be made according to the neighbourhood of points or segments by interpreting the situation on the source map during the encoding process.

The inclusion of additional parameters into the ULR data set is guided by a set of heuristics which use both the original location at hand and the region in the source map where it is situated, including additional map attributes such as street categories of the segments or physical heights or levels in a layer model. Also, particular implementations are free to omit the

[6] this is further described later on in this document

generation (producer-side) or interpretation (consumer-side) of certain optional parameters, depending on their requirements w.r.t. matching accuracy vs. computational effort or complexity of the implementation. This explicitly includes the possibility of a simplistic client implementation which only interprets the mandatory ULR parameters and completely ignores the advanced disambiguation techniques described further down this paper.

Decision items

- Segment or point with similar characteristic in the neighbourhood (direction AND category)
- Roundabout is involved (may lead to problems with different maps, map versions or simplified and lossy map representations on the consumer side)
- Very short distance of the current point to other points on the source map (may lead to conflicts with offsets in the destination map and therefore wrong segment selection)
- Relations of connected segments (will support a more tolerant map interpretation within a given message scope)
- Some applications or events may directly implicate more parameters (see the parameters in the following Area example)
- ... (more may be suitable)

If this decision process suggests additional describing parameters for the mapping process of the current point to the segment on the source map the ULR Markov algorithm data is generated and included into the ULR data set. Some of the mentioned parameters are mandatory, some are optional. This algorithm uses some of the other ULR parameters as well, i.e. the co-ordinates.

Co-ordinate options

Over all location types each pair of involved co-ordinates may be enriched by:

- ULR Markov parameters
- Level (stacking model with description)
- Accuracy (of co-ordinates)

Location type options

- For Point:
  - Adding a point descriptor (not handled herein)
- For Linear:
  - Adding a linear descriptor (not handled herein)
- For Area:
  - For a simple area: Setting the expansion
  - Setting inner points inside the area (e.g. sub-areas inside an parking area, rescue location of special event areas etc)
  - Determine any additional parameters needed for characterizing the area or sub-area and place them in the ULR data set.
  - Adding an area descriptor (not handled herein)

ULR Markov algorithm parameters

This algorithm is especially useful to identify to originally involved segment out of similar segments in the neighbourhood. The algorithm itself is optional and most of its parameters are optional as well. Even the typical Markov result vector may be omitted if the simplified distance parameters seem to provide enough disambiguation. So the length of this structure varies. The Markov algorithm will use the segment direction and normalized category.

- valMode1: coded mode for the Markov processing environment
- radius: circle definition of the involved map area around the current point
- val1: recurrence frequency 1
- val2: recurrence frequency 2
- valMode2: additional parameter for the Markov processing environment if applicable
- fromPoint: start point of the involved segment in the direction to the current point
- toPoint: end point of the involved segment, coming from the current point

The usage of the ULR Markov data set is optional and it enhances the probability to correctly match the event or point on the client side. For devices with limited resources a simple match purely based on the WGS84 co-ordinates and direction is always possible. If fromPoint and/or toPoint exist in the data set, a reduced interpretation of these values is possible as well. This holds true for all other additional information in the data set.

In ULR, a linear location may express not only a sequence of segments but, optionally, relations between adjacent segments in the sequence. This way, a location can describe limitations in the *combined* usage of these segments (e.g., turn restrictions). There are provisions to mark this situation in the ULR data set. This is important for the producer side to generate a map deviation-tolerant description and to support the Markov algorithm in otherwise difficult-to-handle situations (e.g. short segments in two-lane crossroads and an adapted simplification for map deviation-tolerance on the consumer side).

The Markov algorithm itself will take some computation time because of its iterative character. In most cases it can be assumed to invoke this process only if the co-ordinate-based interpretation indicates a direct involvement of the event within the own route. So on the other hand in most cases it can be expected that with such a skipping no additional computation time will be needed and the client system therefore wouldn’t be slowed down at all.

ULR Markov Algorithm

This section introduces the new concepts for the basic problem of identifying individual road segments between two (slightly) different maps. In the following, segment always means a directed segment of a road (or a similar line-shaped map feature).

The first of the following two subsections introduces the concept for the new segment attribute, based on the recurrence frequencies of a certain Markov chain. The second one summarizes all the aspects that manifest themselves in the set-up of the transition matrix of the Markov chain.

Disambiguation of Segments

The basic idea is to devise new segment attributes that are derived not only from information attached to the segment itself, such as co-ordinates and road category, but from an entire neighbourhood of the segment in an algorithmic manner. Values of these attributes then contribute to a more accurate identification of segments on the consumer side. The new attribute is defined in such a way that it is as robust as possible under changes of co-ordinates and network topology.

In an example realization, the following settings are used:

- A circular neighbourhood around the current point[7] is determined such that it contains a certain number of segments (e.g. between 80 and 120[8])
- For the resulting network, the traffic density is computed according to certain rules (more details on this below)

Note that the traffic density concept employed doesn't have to be realistic, but it needs to be deterministic, robust, and expressive.

The following traffic density concept is employed:

- A Markov chain is set up
- Its states are the (oriented) segments within the neighbourhood of the current point

The set-up of the transition probabilities is described in the following section.

It will be ensured that the Markov chain possesses exactly one equivalence class of communicating states, which suffices to establish that the vector of normalized recurrence frequencies (the steady distribution) is unique. These normalized recurrence frequencies constitute the values of the new segment attribute.

As to the computation of this vector of recurrence frequencies, it can be obtained by taking the left eigenvector of the transition matrix corresponding to the eigenvalue 1.0 (which always exists and is simple by construction) with subsequent normalization. Since 1.0 is the dominant eigenvalue of any stochastic matrix, the power method constitutes a simple means to compute the corresponding eigenvector.

Transition Probabilities

Most details of the entire procedure lie in the set-up of the transition probabilities between the segments.

Because the resulting recurrence frequencies are required to provide some robustness against co-ordinate deviations and small topology changes, the same requirement holds for the transition probabilities. Hence, the transition probabilities are assigned with a fuzzy approach as follows:

[7] „current point“ in the sense of the Markov algorithm is the point that should be matched between producer and consumer maps

[8] This is a practical number and it can be varied for enhancements

- Generally, factors (weights) influencing the transition probabilities that are based on co-ordinates, lengths, etc. use continuous interpolations between discrete extrema (instead of discrete yes/no decisions).
- As a concrete example, short segments, or sequences of consecutive short segments, can be "tunnelled" with a weight that continuously degrades to zero as the segment length increases.

The steps in the construction of the transition matrix are as follows:

- All entries are set to a small positive number; the value may continuously depend on the co-ordinates involved.
- For any two consecutive segments (i.e., the second one begins at the same node where the first one ends), and for any two segments that are connected by short other segment(s), a large transition weight is set, which continuously depends on the distance from the current point and the distance between the end point of the first segment and the beginning point of the second segment, as described above.
- This aforementioned transition weight is further decreased for turns, down to zero for U turns; however, the dependency on the turning angle is continuously reduced to zero below a certain length of the involved segments, so as to avoid discontinuities at the origin.
- The aforementioned transition weight is also made dependent on the category (importance) of the second segment: Turns into roads of lower category are slightly lower weighted than turns into roads of higher category.
- Finally, each row of the matrix is normalized such that its sum is 1.0.

This concludes the construction of the stochastic transition matrix. Observe that all entries are positive, which ensures the uniqueness of the steady distribution.

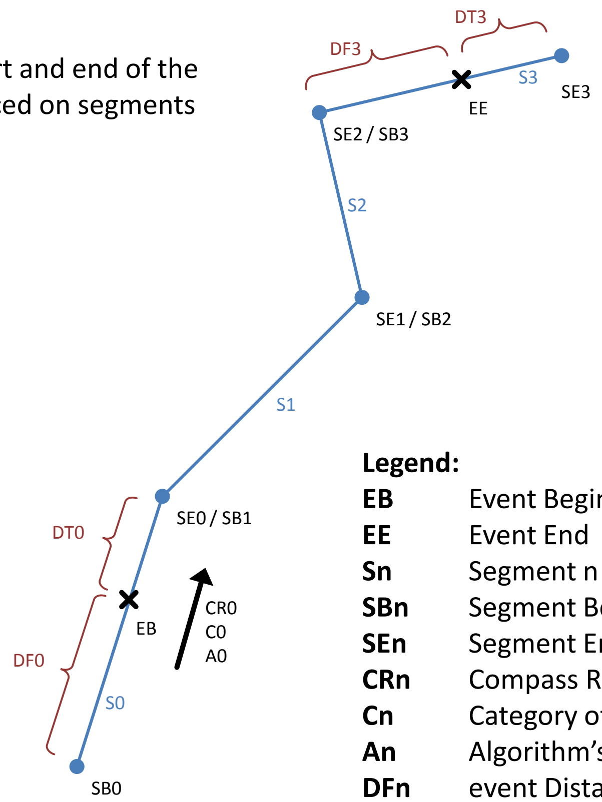
Examples

1. Linear example with descriptive parameters

Figure 1 shows relevant attributes with shortcuts of a linear ULR representation.

It is a complex linear location with intermediate (via) points and constructed with sequentially ordered segments, each with a different direction.

Example:
Polyline with start and end of the event, freely placed on segments



Legend:

EB	Event Begin
EE	Event End
Sn	Segment n
SBn	Segment Begin of segment n
SEn	Segment End of segment n
CRn	Compass Rose of segment n (direction)
Cn	Category of segment n
An	Algorithm's dataset of segment n
DFn	event Distance From begin of segment n
DTn	event Distance To end of segment n

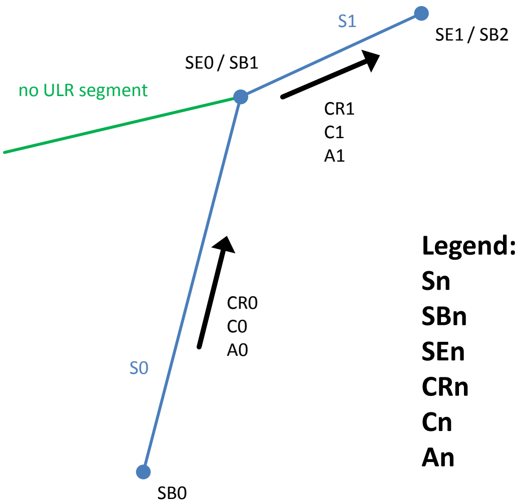
Figure 1 Linear example

2. Linear example with descriptive parameters and relations

Figure 2 shows relevant attributes with shortcuts of a linear ULR representation with relations between segments.

It is a linear location with an intermediate (via) point and constructed with sequentially ordered segments, each with a different direction. A special application demonstrates the handling on client side. Both segments of the example are in relation and so the according linear event is relevant to S0 and all further members. Interpretation of the event at the entry point SB1 is of no relevance although S1 is on the own path. "event" may be e.g. a blocked route in exactly this construction (coming from S0, passing S1).

Example:
Linear, relations between segments,
A1 will include the relation,
a „no ULR segment" uses SB1 as side-entry



Legend:

Sn	Segment n
SBn	Segment Begin of segment n
SEn	Segment End of segment n
CRn	Compass Rose of segment n (direction)
Cn	Category of segment n
An	Algorithm's dataset of segment n

Figure 2 Linear with relations

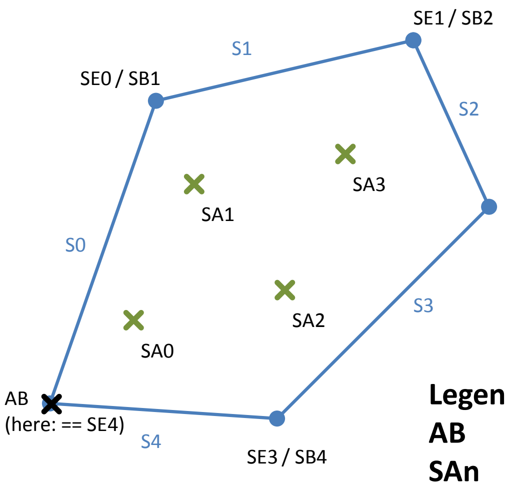
3. Area example with descriptive parameters

Figure 3 shows relevant attributes with shortcuts of a complex area ULR representation.

The complex area is surrounded by sequentially ordered segments, where the first segment starts at the start co-ordinate and the last segment will close the polygon at this start co-ordinate.

Furthermore some inner co-ordinates may describe local subsets inside the primary area, e.g. the occupancy rate of named parking zones.

Example:
Parking area with a primary co-ordinate
and further sub-areas, e.g. inner parking zones



Legend:

AB	Area Begin
SAn	Sub-Area n
Sn	(Polygon-) Segment n
SBn	Segment Begin of segment n
SEn	Segment End of segment n

Figure 3 Complex area

4. ULR Markov algorithm based match

The following figures demonstrate a segment placed on the map of the producer side (Figure 4, segment in red) and the intended equivalent segment on the client-side (Figure 5, segment in red) within a different map. Characteristics are near-by co-ordinates (as seen on producer side) and topology changes on client side.

The new segment attributes are helpful for the identification process.

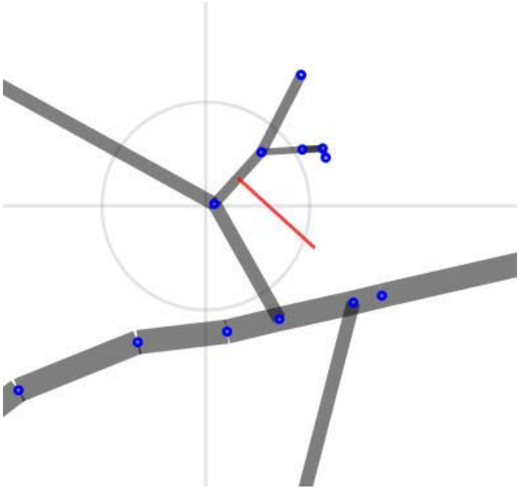


Figure 4 Producer-side mapped segment

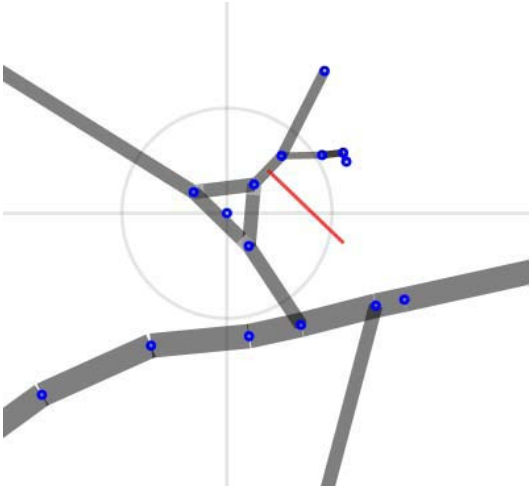


Figure 5 Client-side identified ULR object

References

[1] TPEG – Transport Protocol Experts Group; http://www.tisa.org/technologies/tpeg/

[2] TISA – Traveller Information Services Association; http://www.tisa.org/

[3] TMC – Traffic Message Channel; http://www.tisa.org/technologies/tmc/

[4] RDS-TMC (ISO 14819 Series); http://www.iso.org/

[5] TPEG Binary Specification – Generation 1 (CEN/ISO TS 18234 Series)

[6] tpegML (XML) Specification (CEN/ISO TS 24530 Series)

[7] TPEG2 – Generation 2 (prCEN ISO/TS 21219 Series)